TECHNICAL REPORT 3047
September 2016

Radio Model-free Noise Reduction of Radio Transmissions with Convolutional Autoencoders

Benjamin Migliori, Ph.D.
Daniel Gebhardt, Ph.D.
Logan Straatemeier
Michael Walton

SSC Pacific
San Diego, CA 92152-5001

SSC Pacific
San Diego, California 92152-5001

K. J. Rothenhaus, CAPT, USN
Commanding Officer

C. A. Keeney
Executive Director

ADMINISTRATIVE INFORMATION

The work described in this report was performed by the IO Support to National Security Branch (Code 56120), the IO Spectrum Exploitation Branch (Code 56140), and Exploitation Systems Branch (Code 56150) of the Information Operations Division (Code 56100), Space and Naval Warfare Systems Center Pacific (SSC Pacific), San Diego, CA. The Naval Innovative Science and Engineering (NISE) Program at SSC Pacific funded this Applied Research project.

Released by
B. Dixon, Head
IO Support to National Security Branch

Under authority of
G. Settelmayer, Head
Information Operations Division

SB

EXECUTIVE SUMMARY

This technical report presents the findings of an experiment designed to evaluate the effectiveness of our proposed method to remove arbitrary forms of signal contamination via unsupervised machine learning. In real-world situations, electromagnetic transmissions are affected by the propagation environment in which they are transmitted. While certain phenomena (i.e., ducting) can be beneficial, many types of propagation effects are deleterious and reduce the ability of the end receiver to successfully decode the signal. These effects include various types of environmental effects (multipath interference, fading, etc.), but may also include active effects, i.e., jamming. Much of the effort in improving the signal-to-noise ratio (SNR) of radio-frequency (RF) transmissions resides in the physical layer, i.e., antennas, cabling, and filters. These physical layer tools have an established body of work and solid theoretical backing; however, by virtue of requiring physical changes they are not as adaptable to changing environments as signal processing models. Digital signal processing methods (DSP) are used in an attempt to boost signal-to-noise ratio without altering the physical layer. This method can be as simple as using matched filtering, in which a time-reversed template signal is convolved with the unknown signal to locate the maximum correlation, or Fourier-based filtering, in which certain spectral bands are suppressed to excise the signal. More complex methods can also be used, such as Bayesian decision rules to decide if a signal is present or not present (optimal detection theory). Sometimes, representing the signal in a different way, such as by projecting into a new space in which the signal and noise no longer overlap (via wavelet or other decompositions) can allow an improvement in SNR by allowing separation of signal and noise.

A crucial weakness of these "standard" methods of improving SNR is the reliance upon prior information about the input signal to the denoising method. "Blind" SNR improvement is much more challenging. Many signals have both high- and low-frequency components, and they are often mixed in complex ways. The same can be said of both passive and active signal contamination. Without a clear knowledge of the specific features of the signal and the noise, blind methods are very likely to filter out important features of the signal. Similarly, it is insufficient to have a known signal and an unknown noise, or vice versa; again, blind filtering is likely to have detrimental effects on the signal.

We identify a machine learning method for removing noise contamination of signals without advance knowledge of the signal model, noise model, or exact propagation model. We accomplish this by using unsupervised machine learning implemented with regularized convolutional autoencoders and "probe" signals, a series of exemplars containing randomly encoded information and transmitted within the relevant propagation environment. We show, for multiple modulation schemes, that previously unseen contaminated signals can be boosted by over 6 dB when contaminated with additive white Gaussian noise.

Although in this report, we utilize known modulations and known noise for our dataset, at no point is this information communicated to the denoising system. Thus, our system demonstrates adaptation to the environment and the capability to adapt as the environment changes. We recommend further exploration of this method, specifically in the presence of more realistic noise models and in the presence of active noise sources.

This work was done as part of the BIAS (Biologically Inspired Autonomous Sensing) project, funded from the Naval Innovative Science and Engineering (NISE) Program.

CONTENTS

Figures

Tables

1. INTRODUCTION

The desire to successfully transmit information in the presence of complex noise sources has been considered from the earliest days of wireless transmission; it stands as a crucial component for radio-frequency (RF) system design.

In real-world situations, electromagnetic transmissions are effected by the propagation environment in which they are transmitted. While certain phenomena (i.e., ducting) can be beneficial, many types of propagation effects are deleterious and reduce the ability of the end receiver to successfully decode the signal. These effects include various types of environmental effects (multipath interference, fading, etc.), but may also include active effects, i.e., jamming.

Much of the effort in improving the signal-to-noise ratio (SNR) of RF transmissions resides in the physical layer, i.e., antennas, cabling, and filters. These physical layer tools have an established body of work and solid theoretical backing; however, by virtue of requiring physical changes, they are not as adaptable to changing environments as signal processing models. We will not consider physical layer methods further in this report, and will act under the assumption that all such methods have been employed to maximum effect on the system being studied.

Analog or digital signal processing methods are used to boost SNRs without altering the physical layer. These fall, broadly, into three categories: information-theoretic methods, filter-based methods, projection-based methods. Each of these methods has a large number of related techniques [11]. Briefly, information theoretic methods [6] utilize Bayesian statistics (maximum a posteriori probability, for example) to create an optimal estimate for the source signal in the absence of noise contaminants. Filter-based methods, such as matched filtering (in which a time-reversed template signal is convolved with the unknown signal to locate the maximum correlation), or Fourier-based filtering (in which certain spectral bands are suppressed to excise the signal) can be applied adaptively or non-adaptively. Each of these methods require significant a priori knowledge of the signal model And the noise model, without knowledge of symbol type and transmission characteristics,, information-theoretic methods cannot be applied and filter-based methods cannot be specified to the specific signals under investigation.

Projection-based methods, ranging from Fourier or wavelet decomposition [23] to independent component analysis (ICA) [2], are of particular interest as they do not necessarily require advance knowledge of the signal model or the noise model. Instead, the projection basis functions can be chosen to maximally separate an unknown noise source from a series of example signals, which is equivalent to projecting into a space where the signal and the noise no longer overlap. This approach provides a significant advantage; however, determination of the optimal projection basis or decomposition functions requires substantial analysis. Similarly, the wavelet thresholds must either be selected manually or they must assume Gaussian noise distributions [18] [23].

A crucial weakness of these "standard" methods of improving SNR is the reliance upon prior information about the denoised signal. "Blind" SNR improvement is much more challenging. Many signals have both high- and low-frequency components, and they are often mixed in complex ways. The same can be said of both passive and active signal contamination. Without a clear knowledge of the specific features of the signal and the noise, blind methods are very likely to filter out important features of the signal. Similarly, it is insufficient to have

a known signal and an unknown noise, or vice versa; again, blind filtering is likely to have detrimental effects on the signal.

Unsupervised learning presents a possible solution and is uniquely suited for the challenge of denoising non-adversarial signals in an unspecified propagation environment. Unsupervised learning [3] learns important features and representations directly from unlabeled example data, and can then generalize what has been learned to new data. In the denoising context, this can be thought of as a mixture of data-adaptive filtering methods and projection methods. Generally, unsupervised learning is limited by the quality of the datasets available; for a new propagation environment, one would not usually have an adequate dataset available for unsupervised learning. For the problem of denoising non-adversarial signals, however, it would be feasible to send random symbols via an arbitrary modulation within a dynamic propagation environment. Random transmitted information would prevent adversarial information gathering attempts, and it would provide a call-and-response set of propagation examples. These examples could serve as "probe" signals, and could form the training set needed to learn an inverse transformation from noisy signals to clean ones.

We demonstrate unsupervised learning of this inverse transformation under signal contamination with additive white Gaussian noise. We show that the signal-to-noise ratio can be improved by an average of over 6 dB, corresponding to a fourfold SNR improvement under certain circumstances, which induces a corresponding factor of 2 increase in the noisy channel capacity, with no further optimizations.

2. METHODS AND EXPERIMENT

We will use the terms *instance* or *example* to refer to a particular input $x \in X$. A *sample* may refer to either a randomly drawn subset $\hat{X} \subseteq X$ or a single measurement x_t in a particular instance and should be clear from context. Because much of the work in convolutional neural networks (CNNs) has been conducted in the computer vision domain, much of the terminology and notation assumes $c \times w \times h$ input where c is the color channel or feature dimension[1] and w, h are the two dimensional spatial resolution of the input image. In the temporal radio frequency domain, our signals are $c \times s$, where c is the number of channels and s is the number of samples per instance. Unless otherwise indicated, we define all operations (convolutions, downsampling, normalizations, etc.) to operate across the last dimensions of the input signals, treating the c dimension as independent, which implies spatial and sample-wise operations in the image and RF domains respectively.

We present an application of information-preserving convolutional autoencoders (also described as deconvolutional autoencoders [25]) capable of removing arbitrary noise $n(t)$ from a contaminated signal, $s(t)$. To gauge the performance of our system, $\psi \circ \phi$, we ingest a contaminated signal and produce a prediction of $s(t)$, i.e., $\psi \circ \phi(s+n) \rightarrow s$. The performance of the system may then be gauged by a loss function between a known test signal s' contaminated with additive white Gaussian noise (AWGN, n') and the prediction $\psi \circ \phi(s' + n')$.

2.1 CONVOLUTIONAL AUTOENCODERS (CAE)

In its most general form, an autoencoder is composed of an encoder network ϕ that maps inputs X to a latent space Z and a decoder network ψ that attempts to reconstruct the input. Typically, the objective function to be minimized by this class of models is simply the squared error between the input and its reconstruction. This is often referred to as the "reconstruction loss" $L_{rec} = ||X - \psi \circ \phi(X)||_2$, where L_{rec} is $||x||_p$ is the p-norm of a vector x. For a discussion on traditionalautoencoders as applied to radio signals, we advise the reader to refer to the Space and Naval Warfare Systems Center Pacific technical report on the subject [16].

A convolutional autoencoder (CAE) is simply an autoencoder that incorporates model design principals from CNNs [14]. The encoder path of a CAE (ϕ) is a standard CNN; the structure of the decoder, however, is non-trivial. In classical autoencoders, an appropriate definition for the decoder path (ψ) follows naturally by inverting the matrix multiplication operations at each layer in a "mirrored" fashion. This is not possible for traditional CNNs, as some operations (such as max pooling) are lossy and non-invertible. Here, we briefly cover the techniques required to enable convolutional autoencoders to encode information endto-end, rather than in a layerwise fashion. For a more detailed discussion, we advise the reader to refer to the SSC Pacific technical report [15] in review at the time of publication.

2.1.1 Inverting Max Pooling

Pooling operations subsample a signal to produce a single output from subregions of the input. Naive approaches to reversing the max pooling operation by interpolation have been explored extensively in [1] [26].

[1] If the input to a particular operation is X, this is often referred to as a "channel"; if instead the input to an operation is the output of some mapping, this may be referred to as a feature

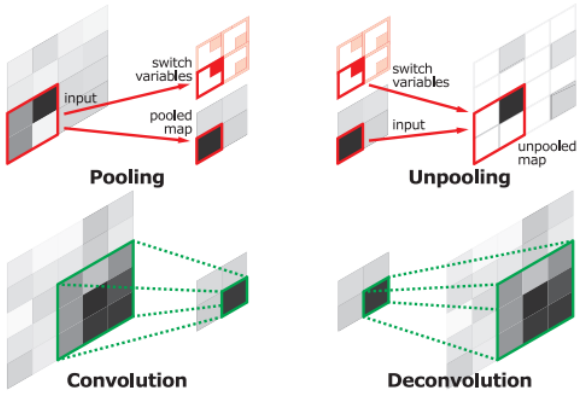

Figure 1. A diagram describing the crucial features of convolutional autoencoders. Pooling and convolution condenses information on the ϕ pass of the autoencoder. On the ψ pass, deconvolution allows reconstruction of the input from knowledge of the convolutional kernel and the output. Similarly, unpooling uses switch variables to recreate an unpooled map from a single input. Importantly, the unpooling stage is not lossless, and only retains information corresponding to the maximum feature present.

A natural extension to max pooling is to define an operation that returns the *argmax* of each pooling region; that is, the i, j coordinates of each maxima of a feature in the encoder in response to a particular input. To invert the max pool, it is then straightforward to construct a representation of the same dimension as the input with the maxima inserted at coordinates i, j, and zeros elsewhere. This approximate inverse of the pooling operation, commonly referred to as "unpooling" [24], produces a sparse upsampling as in [7] without discarding structural information by arbitrarily inserting the maxima. In the literature, the coordinates of the maxima are commonly referred to as *switches* or *switch variables*; for consistency, we will use these terms as well. This process is shown in the upper half of Figure 1. An example of the difference between simple upsampling and unpooling is shown in Figure 2.

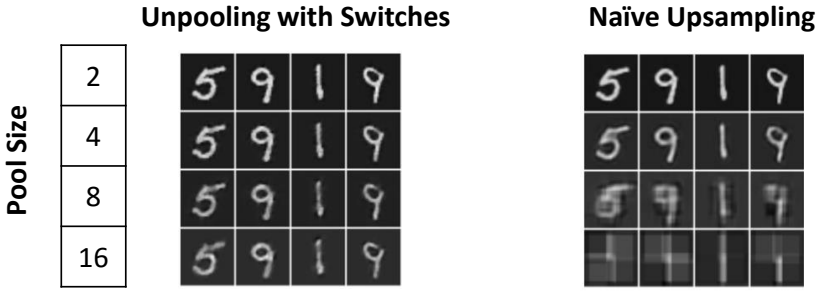

Figure 2. A diagram showing the effects of unpooling versus simple upsampling. Upsampling results in significant loss of information.

2.1.2 Inverting Convolutional Layers

The notion of reversing convolutions was first introduced in [25] and termed a *projection operation* in [24]. Note that the filters used in the decoder may have its own parameters, as in [26], or share weights with the corresponding filter in the encoder (harkening back to the notion of weight-tying in traditional autoencoders), as in [25] [24].

The mechanism for properly inverting the convolution operation of a CNN is most clear in a matrix representation. In the simple case of a square input with size $n = 3, 3$, we begin by flattening the input into a vector $v = vec(x) \in \mathbb{R}^{n \cdot n}$. For a convolutional kernel of size $k = 2, 2$, stride of $s = 1$, and padding of $p = 0$, the output of this convolution will be of size $2, 2$. Thus, we can arrange a matrix with non-zero elements corresponding to the w_{ij} components of the convolution filter as

$$W = \begin{bmatrix} w_{00} & w_{01} & 0 & w_{10} & w_{11} & 0 & 0 & 0 & 0 \\ 0 & w_{00} & w_{01} & 0 & w_{10} & w_{11} & 0 & 0 & 0 \\ 0 & 0 & 0 & w_{00} & w_{01} & 0 & w_{10} & w_{11} & 0 \\ 0 & 0 & 0 & 0 & w_{00} & w_{01} & 0 & w_{10} & w_{11} \end{bmatrix}. \tag{1}$$

Now, convolution with the desired kernel becomes a simple matrix multiplication $v \cdot W$, which produces a vector $v' \in \mathbb{R}^4$. Reshaping v' into the desired $2, 2$ shape yields the convolved output. Thus, the corresponding deconvolution is in fact W^T. Further, as observed in [9], the transpose convolution is the gradient of the corresponding convolution (satisfying certain conditions) with respect to its input. In many modern machine learning libraries, the "forward pass" of a convolutional layer is computed using W and the "backward pass" (used for back-propagation) is computed by W^T. In practice, the transpose convolution may be implemented by simply swapping the forward and backward pass. This process is shown in the lower half of Figure 1.

2.2 SYNTHETIC SIGNALS

The test and training dataset consists of synthetically generated radio signals clean of outside interference. We used the GNU Radio [4] software-defined radio (SDR) framework to construct the modulations that generated this data.

A binary file, produced by randomly choosing byte values $[0, 255]$, is the waveforms' input. These binary data are modulated as in-phase and quadrature I/Q samples using each of six methods: on-off keying (OOK), Gaussian frequency-shift keying (GFSK), Gaussian minimum-shift keying (GMSK), differential binary phase-shift keying (DBPSK), differential quadrature phase-shift keying (DQPSK), and orthogonal frequency-division multiplexing (OFDM).

For each modulation, the samples are sent to a Nuand™ BladeRF™ software-defined radio (SDR), where they are upconverted to the carrier frequency. The SDR is configured in RF loop-back mode, such that the RF signal is sent and received only within the device's circuitry, and not to an external antenna. This arrangement provides added realism by incorporating the upconversion and radio effects, but without unwanted third-party signals that could pollute the controlled testing.

The signal sampling rate is set so that the number of samples per symbol (N_{SpS}) is consistent for every modulation type, except for OFDM. In contrast with the other modulation techniques, OFDM encodes data on multiple carrier frequencies simultaneously, within the same symbol, and modulates each carrier frequency independently. Our experiment used an existing OFDM signal processing component that operates with a symbol rate different than the other configurations, but with the same sample rate. This rate is identical for both the transmission and reception of the signal. The received RF signal is down-converted at the radio and the resulting I/Q samples are stored for analysis.

The data files need to be arranged into a format and structure for use by our neural network. The I/Q data are split into segments consisting of N_{SpV} samples, or samples per vector. A segment is composed of interleaved I and Q values for each sample, forming a vector of length $2 \times N_{SpV}$. Thus, each vector contains $\frac{N_{SpV}}{N_{SpS}}$ symbols. These vectors are placed into two sets, *train* and *test* (sizes N_{Vtrain} and N_{Vtest}), such that both the modulation type and positions within the set are random. The parameter N_{SpV} is identical for each modulation type for all experiments described in this report. The specific values of all parameters are shown in Table 1.

Table 1. Data generation parameters.

Description	Parameter	Value
Samples per symbol	N_{SpS}	10
Samples per vector	N_{SpV}	225
Number of training vectors	N_{Vtrain}	60000
Number of training vectors per modulation	N_{Vmod}	10000
Number of test vectors	N_{Vtest}	10000

Example vectors with and without noise are shown in Figure 3.

2.3 ADDITION OF NOISE

To assess the performance of our system with a more realistic channel model, we altered the test data set with additive white Gaussian noise (AWGN). AWGN was added to each set of signal modulation types such that for each set, the resulting SNR matched a given value. For each of these signal modulation sets, $\{S_{mod}\}$, the added noise power, P_{noise} is

$$P_{noise(mod)} = \beta \cdot \frac{1}{N_{s(mod)}} \sum_{\{S_{mod}\}} \frac{1}{\tau} \sum_{t=1}^{\tau} [s_t]^2, \tag{2}$$

where $N_{s(mod)}$ is the number of sample vectors for a particular modulation, s_t is an individual signal sample vector of length τ, and β is a factor chosen such that $10\log(P_{\{S\}}/P_{noise})$ matches the desired SNR. Examples of modulation data with the addition of noise are shown in Figure 3. Note that all modulations exhibited similar transmitted power, with the exception of OOK, which was slightly larger.

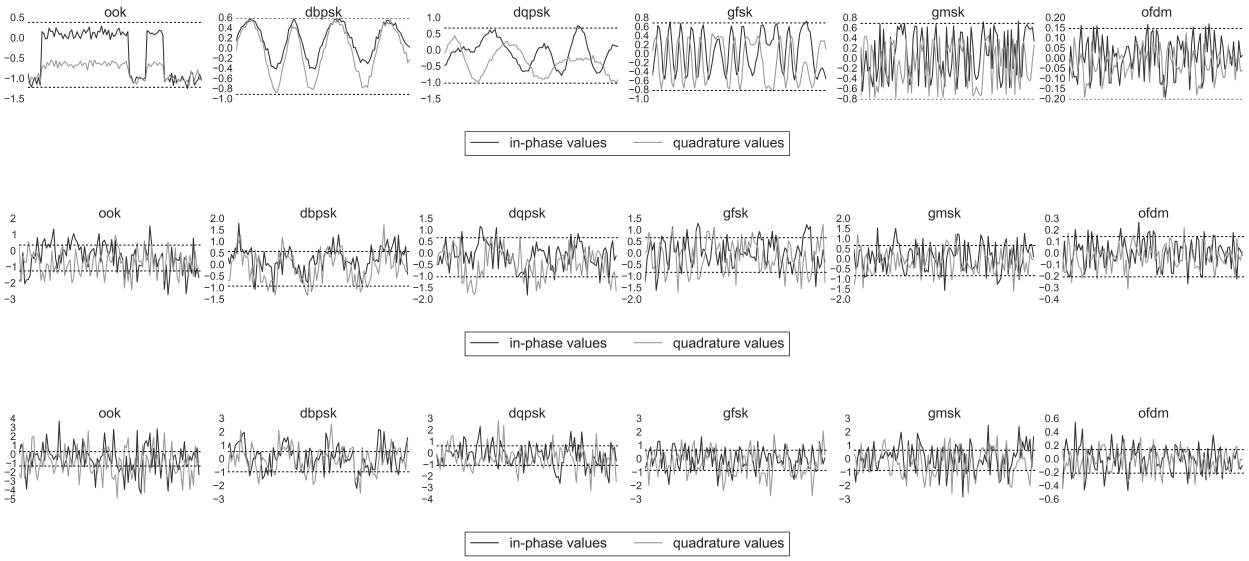

Figure 3. Example input vectors for each modulation type under varying noise. Each instance has 100 samples (I and Q pairs). Dashed lines indicate the envelope of the 20-dB SNR signals. The top panel has an SNR of 20 dB, the middle panel an SNR of 0 dB, and the bottom panel has an SNR of -6 dB. The examples are the same across panels.

2.4 CAE ARCHITECTURE TRAINING

Prior to training, the synthetic dataset was partitioned four equally sized disjoint partitions and shuffled. Each partition was assigned an SNR of 0 dB, 5 dB, 10 dB, or left uncontaminated. Thus, the test and training dataset ($s' + n'$) consists of a randomly distributed application of noise across examples and modulation types, but without the same example appearing both with and without noise. A mirroring dataset was created to track the original signal prior to addition of noise (s'). Each example is normalized based on the RMS power of the modulation class, such that the amplitude of $s' + n'$ is constrained to be approximately between $+1$ and -1. This causes the actual magnitude of s' to vary across examples, preventing learning of amplitude-only features.

For this initial work, we explored two CAE architectures, both using unpooling and deconvolution in the ψ pathway. Batch normalization and dropout were used to improve regularization and prevent overfitting [21]. Each network is trained end-to-end using adaptive gradient descent (Adagrad [8]) until convergence. The loss function was a measure of 'mean square reconstruction loss' $Ł_{rec} = \frac{1}{n_t}\sum_i^n (s_i - \psi \circ \phi(s_i + n_i))^2$. The best performing iteration was kept for further analysis. All simulations were performed in Keras [5].

The training input was set to the contaminated dataset ($s' + n'$) and the reconstruction target (used to compute loss) was chosen to be the clean versions of each example in $s' + n'$, i.e., s'. This approach causes the autoencoder to learn a denoised reconstruction from a variety of noise inputs, rather than a simple reconstruction.

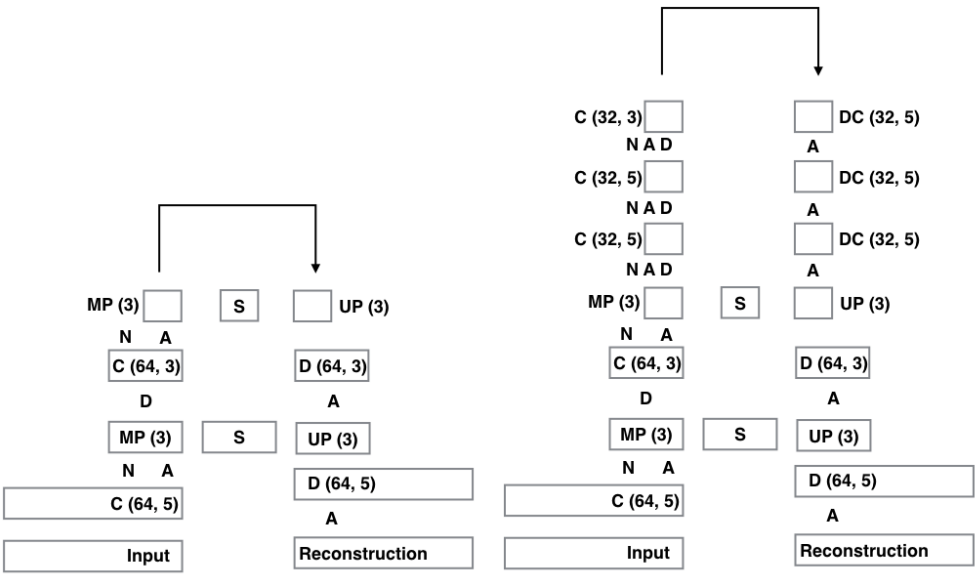

Figure 4. A diagram describing the two architectures used for denoising in this technical report. The left architecture is the 2-layer, and the right is the 5-layer. For each, in the ascending ϕ pathway, *C()* indicates a convolution, with the number of filters and convolution size specified. *N* indicates batch normalization, *A* indicates $tanh$ activation, *D* indicates dropout with a drop probability of 0.25, *MP* indicates maxpooling with the pool size specified, and *S* indicates the switch variables to be passed to the ψ pathway. On the descending ψ pathway, *DC* indicates the deconvolution operation acting as the inverse of the convolution at the same level in the ϕ pathway. *UP* indicates unpooling with the switch variables passed parallel to the two pathways. Note that all operations are done on 2-channel data.

2.4.1 CAE Architectures

2.4.2 K-fold validation

To compensate for potential outliers in network performance caused by specific, random initializations of network weights, stratified k-fold cross validation was performed with $k = 5$. The dataset was partitioned into folds of size $\frac{n_{samples}}{k}$, with equal numbers of samples from each labeled class in each fold. Using a fixed initialization seed, we initialized random weights on five separate models of the 2- and 5-layer architectures. Each of these 10 models was then trained to convergence against a fold. We include individual and ensemble performance in our results.

3. PERFORMANCE CHARACTERIZATION

For each signal $s' + n'$, we used the CAE architectures described above to emit a prediction for s'. To compute the performance of a given architecture, the mean per-sample squared error per channel is computed as

$$MSE_c = \frac{1}{N_s} \sum_{i}^{N} \sum_{t}^{\tau} \left(s(t)_i^c - \psi \circ \phi \left[s(t)_i^c + n(t)\right]\right)^2, \tag{3}$$

where N_s is the number of samples in the test set, τ is the number of timesteps in each sample, c is the channel (I or Q), and n is the AWGN added, if present. This target is used as the loss function for the CAE architectures shown in Figure 4.

One consequence of using normalized inputs with noise added stochastically is that the actual signal amplitude within the test and training set is obscured; because all signals are normalized to unity RMS power prior to processing, a test example at 0 dB will have the original signal attenuated by a factor of two within the noise background, which somewhat complicates the computation of the shift in SNR ratio.

We begin by computing the mean of all RMS noise power levels added to the dataset. As described before, our probe dataset contains a disjoint mix of noise levels, artificially added to reach a target SNR. For 25% of the data, no noise is added; for the remaining 75%, noise is added such that the mean SNR is 3.75 dB. Given initial signals with a normalized RMS power of 1, to reduce the SNR to 3.75dB requires the addition of 0.42 units of normalized RMS noise (i.e., slightly less than half of the signal power), which also implies that the mean normalized signal amplitude within the probe dataset set is, similarly, 0.42. In the case where no noise was added and our CAE produced a perfect reconstruction, it is easy to see that $s - \phi \circ \psi \left[s + 0\right] = 0$, which is equivalent to saying that no noise was introduced by either propagation or reconstruction.

In the more realistic case where the reconstruction is imperfect and the noise term is not zero, we compute the residual noise as

$$s - \phi \circ \psi \left[s + n\right] = n_{res}. \tag{4}$$

The RMS power of n_{res} will tend to zero for perfect denoising and reconstruction. Similarly, for no denoising but perfect reconstruction, n_res will be equivalent to the original noise power added to the probe set.

In both the ground truth examples (s) and the reconstruction ($\phi \circ \psi(s + n)$), the signals are normalized due to the construction of the architecture. Therefore, they will both have RMS normalized power close to 1. To compute the change in SNR, we compute the RMS power of the residual noise $RMS(n_{res})$ and compare it to the SNR of the probe dataset (0.42 in normalized units).

We also compute the channel capacity according to Shannon's formulation [20]:

$$C = B \log_2 \left(1 + \frac{S}{N}\right), \tag{5}$$

where B is the channel bandwidth, and $\frac{S}{N}$ is the SNR. We compute the relative change in C, thus making the result bandwidth independent.

Table 2. Parameters used for training.

Description	Value
Activation function	$tanh$
Learning rate	0.1
Epochs	100
Epsilon	$1e-08$
Decay	0.0

4. RESULTS

Ten models were explored, five belonging to each architecture. Each model was randomly initialized and run on one of the five folds of the dataset. The learning parameters are given by Table 2. Examples of denoised signals are shown in Figure 5 for both architectures.

4.1 NUMERICAL RESULTS

Table 3. Results for 2-layer CAE architecture.

Description	Value
Mean squared error	0.111 $\pm$0.007
SNR ratio change	2.86 dB
Channel capacity change	+0.27

Table 4. Results for 5-layer CAE architecture.

Description	Value
Mean squared error	0.887 $\pm$0.011
SNR ratio change	3.44 dB
Channel capacity change	+0.49

4.2 GRAPHICAL EXAMPLES OF DENOISING

Our results indicate that convolutional autoencoder denoising can result in significant improvement of the signal-to-noise ratios, without the use of any expert-guided parameter settings. The depth of the network used has an impact on the ability of the system to denoise input signals. As can be seen in Figure 5 and in Tables 3 and 4, using a 5-layer network results in greater reduction of MSE, an improvement in SNR shift, and a corresponding improvement in fractional channel capacity. The best case performance, an increase in SNR of 3.33 dB, was achieved using the 5-layer network.

This result is likely due to the additional representational capacity of the deeper networks. Additional regularization of the model is provided by the switch variables and the end-to-end

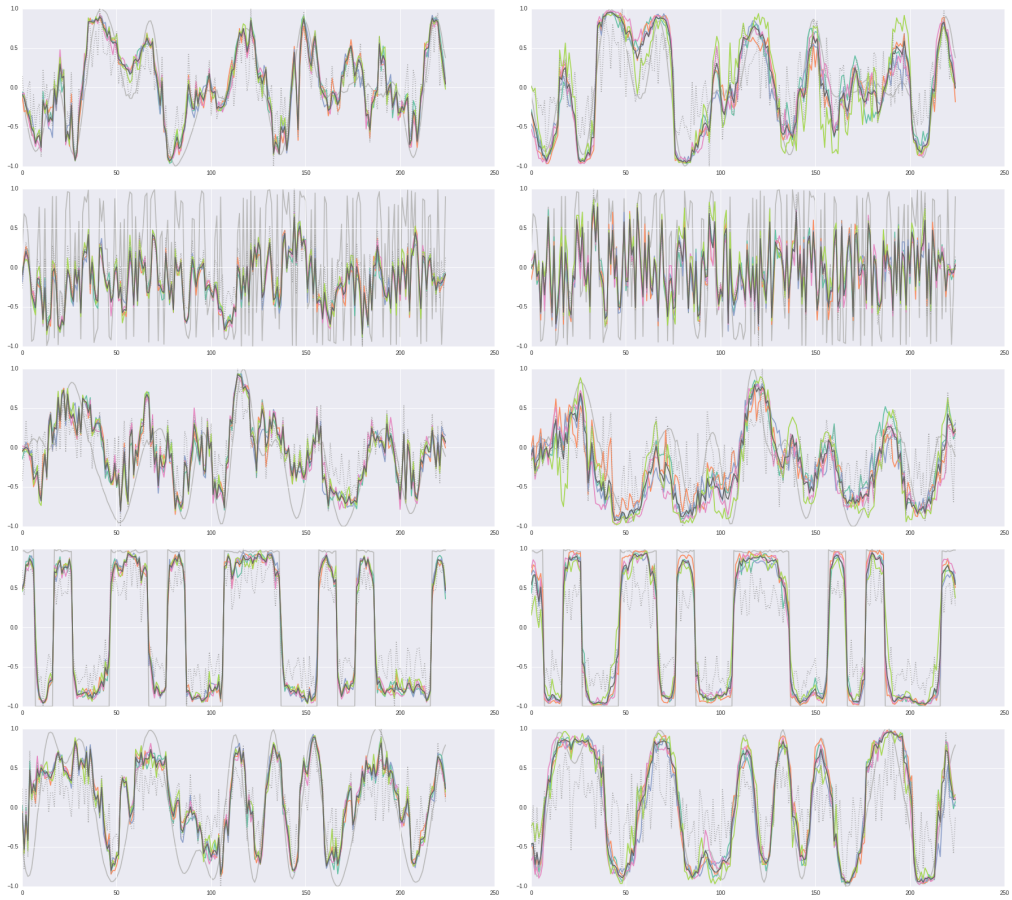

Figure 5. Examples of denoised signals using the 2-layer architecture (left) and the 5-layer architecture (right). The dashed line indicates the contaminated input, the faint solid line the unobserved ground truth, the solid line the ensemble prediction of the denoising network, and the colored lines the prediction based on each fold of the dataset.

projection of information during the reconstruction phase. In an operational capacity, the depth and breadth of the network could be adjusted to balance computational needs and denoising fidelity desired. Caution would be warranted in such a parameter exploration, as expanding the network can also induce overfitting to the test set.

The fidelity of the reconstruction varies depending on the architecture choice and the complexity of the signal. In the second panel from the top, in Figure 5, a GMSK modulation with its characteristic high-frequency transitions is reconstructed. Although neither model can fully remove the noise from this example, the deeper network can track the high-frequency components to a greater extent.

5. DISCUSSION

Our results demonstrate that the SNR of an arbitrary digital modulation may be increased through the use of unsupervised learning, specifically with convolutional autoencoders. This result is important because it represents a "blind" denoising method with a minimum number of parameters to be defined by the user. In the framework of much existing work, our denoising method utilizes learned filters to optimally project into an arbitrary space, while leaving as much of the noise behind as possible, and then projects back into the original space, leaving additional noise behind on the re-projection path as well. Much of the comparable work (wavelet denoising [18], and denoising source separation [19]), have additional requirements, be it number of channels required, thresholds for filtering, or other prior information. The technique introduced here allows denoising of a single recording of a single channel containing fully non-stationary information; we believe this to be one of the few techniques able to tackle such a task without prior knowledge beyond signal examples.

Our proposed method is not without disadvantages. It relies on the transmission (or detection, in the sense of a "natural experiment") of probe signals. These probe signals must be of the same general family as the signals to be denoised, and must be recording in the approximate environment in which the non-probe signals will be transmitted. This would normally be a crucial disadvantage, but for application to denoising a transmission within an uncertain environment, it is feasible to continuously adapt by constantly probing the environment. This probing is almost certainly needed; by taking a purely data-driven approach, the learned denoising transform will not generalize as the environment changes unless learning is continued. Note that our method cannot denoise a signal "out-of-the-box"; it does require training time, and will improve as the duration of the training time increases.

It could be argued that simply sending the intended transmission, recording the arrival, and computing the transfer function would perform in a superior manner (i.e., matched filtering or a host of other similar methods). However, in circumstances (such as certain military transmissions) in which it is desirable to prevent covert observation of a given signal, sending out the intended transmission may not be possible. In such a case, our system provides a "next-best" approach that allows learning of the environmental effects without transmitting crucial information.

It is also important to distinguish our work from the seminal work of Claude Shannon [20] and others on the information capacity of a channel, just as it is important to distinguish it from other types of denoising and source separation. What we have proposed should be viewed as an additional tool in the signal processing pipeline, to be used as part of a chain including many adjustments. By providing a black-box SNR booster, this tool can be employed wherever it makes maximal sense in the pipeline. As an example, given an arbitrary system in which all physical layer, processing, and channel bandwidth adjustments have been made and optimized, a boost in the SNR gives that system additional flexibility to achieve higher channel capacity without sacrificing data rate or making other tradeoffs. Similarly, if source separation is to be performed, our proposed method could be implemented prior to that separation to assist with the removal of uncorrelated information.

Our method also provides distinct advantages. Most methods of improving SNR do not provide information on the characteristics of the noise being removed, primarily because it is not of interest. Autoencoders similarly do not provide direct information on non-represented aspects of a given input. However, because an autoencoder is tasked with making the best

arbitrary reconstruction of a signal, the reconstruction loss provides a performance measure. If this performance measure increases rapidly without a corresponding change in the transmission or receiving system, it indicates that a new phenomena is present. Additionally, it may indicate that this phenomena was not present at the time of training, and thus that an anomaly is present. This could provide guidance for downstream systems tasked with identifying detrimental signals (i.e., jamming) or environmental aspects that should be learned and compensated for.

There are other advantages that should be considered. In the architectures shown in Figure 4, the top layer represents a convolutional encoding of the input. This type of encoding captures useful features for classification [14], description [22], and analogy [13]. Thus, it would be possible to both denoise and encode RF information in a single step. The use of this type of representation for automatic modulation classification is explored in detail in [16, 17]. Additionally, the use of the encoded switch information provided by the CAE may provide an avenue for context-sensitive representation, which may provide useful indicators of system posture or potentially message content based on the relative locations of the features in RF transmissions of different types. The convolution encoding may also prove useful as a front end to a long short term memory (LSTM) [12], which may provide another route towards understanding the content or meaning of RF emissions without direct decoding. For example, an unintended, non-traditional modulation emission could be learned and classified by the system we have proposed. The presence of that emission relative to other events in the spectrum, learnable with an LSTM network, could allow early-warning systems to be semi-autonomously developed with minimal operator intervention. Finally, the use of adversarial networks [10] could be added to the denoising/classification pipeline to improve robustness and generalization ability, especially in the presence of active and known interferers. We recommend further exploration in this area.

6. CONCLUSION

We present in this technical report a machine learning method for removing noise contamination of signals without advance knowledge of the signal model, noise model, or exact propagation model. We demonstrate this result across multiple modulation schemes and noise levels, all contaminated with arbitrary levels of additive white Gaussian noise. We accomplish this by using unsupervised machine learning implemented with regularized convolutional autoencoders and "probe" signals, a series of exemplars containing randomly encoded information and transmitted within the relevant propagation environment.

We show, for multiple modulation schemes, that previously unseen contaminated signals can be boosted by as much as 3.44 dB when contaminated with additive white Gaussian noise. We utilize known modulations and known noise for our dataset, but at no point is this information communicated to the denoising system. Thus, our system demonstrates adaptation to the environment and the capability to adapt as the environment changes. We recommend further exploration of this method, specifically in the presence of more realistic noise models and in the presence of active noise sources.

The method presented here makes advantageous use of its own disadvantages: although probe signals are required, the signals may be random and free of information; although the transforms learned will not generalize, a rapid performance degradation may indicate active interference or major environmental changes. The performance observed in the simple task analyzed here shows promise for application to more challenging scenarios, such as realistic channel models, environments with non-stationary statistics, and active jamming. Additionally, the signal representations required to perform the denoising are highly compatible with cutting-edge machine learning techniques as of the date of this report. We recommend further study in this area.

REFERENCES

1. Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. 2015. “Segnet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation.” Cornell University Library. Computing Research Repository (CoRR). abs/1511.00561.

2. Anthony J. Bell and Terrence J. Sejnowski. 1997. “The Independent Components of Natural Scenes are Edge Filters,” *Vision Research 37*(23):3327–3338.

3. Yoshua Bengio, Aaron C Courville, and Pascal Vincent. 2012. “Unsupervised Feature Learning and Deep Learning: A Review and New Perspectives.” Cornell University Library. Computing Research Repository (CoRR). abs/1206.55381.

4. Eric Blossom. 2004. “Gnu Radio: Tools for Exploring the Radio Frequency Spectrum,” *Linux Journal 122*(June):1–4.

5. Francois Chollet. 2015.“Keras: Deep Learning Library for TensorFlow and Theano.” Available online at https://github.com/fchollet/keras. Accessed February 14, 2017.

6. Donald D. Dorfman and Edward Alf. 1969. “Maximum-likelihood Estimation of Parameters of Signal-detection Theory and Determination of Confidence Interval Rating-method Data.” *Journal of Mathematical Psychology 6*(3):487–496.

7. Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. 2014. “Learning to Generate Chairs with Convolutional Neural Networks.” Cornell University Library. Computing Research Repository (CoRR). abs/1411.5928, 2014.

8. John Duchi, Elad Hazan, and Yoram Singer. 2011. “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization,” *Journal of Machine Learning Research 12* (February):2121–2159.

9. Vincent Dumoulin and Francesco Visin. 2016. “A Guide to Convolution Arithmetic for Deep Learning.” Cornell University Library. Computing Research Repository (CoRR). arXiv:1603.07285.

10. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. “Generative Adversarial Nets.” *Advances in Neural Information Processing Systems: Proceedings of the Neural Information Processing Systems (NIPS) Conference*. December 8–11, Montreal, Canada.

11. Kanika Gupta and S. K. Gupta. 2013. “Image Denoising Techniques—A Review Paper.” *International Journal of Innovative Technology and Exploring Engineering™ (IJITEE) 2*:6–9.

12. Sepp Hochreiter and Jurgen Schmidhuber. 1007. “Long Short-term Memory, *Neural Computation 9*(8):1735–1780.

13. Justin Johnson, Andrej Karpathy, and Li Fei-Fei. 2015. “Densecap: Fully Convolutional Localization Networks for Dense Captioning.” Cornell University Library. Computing Research Repository (CoRR). arXiv preprint arXiv:1511.07571.

14. Yann LeCun, L. D. Jackel, Leon Bottou, Corinna Cortes, John S Denker, Harris Drucker, Isabelle Guyon, U. A. Muller, E. Sackinger, P. Simard, and Valdimir Vapnik. 1995. "Learning Algorithms for Classification: A Comparison on Handwritten Digit Recognition." *Neural Networks: The Statistical Mechanics Perspective: Proceedings of the Center for Theoretical Physics (CTP)-PBSRI Joint Workshop on Theoretical Physics* (pp. 261-276). February 2–4, Postech, Pohang, Korea.

15. Daniel Gebhardt, Michael Walton, Benjamin Migliori, and Logan Straatemeier. 2016. "Learning and Visualizing Modulation Discriminative Radio Signal Features." Technical Report 3048. Space and Naval Warfare Systems Center Pacific (SSC Pacific), San Diego, CA.

16. Benjamin Migliori and Daniel Gebhardt. 2016. "Biologically Inspired Machine Learning for Radio Signal Modulation Classification. Technical Report 3025, Space and Naval Warfare Systems Center Pacific (SSC Pacific), San Diego, CA.

17. Timothy J. O'Shea and Johnathan Corgan. 2016. "Convolutional Radio Modulation Recognition Networks." Cornell University Library. Computing Research Repository. abs/1602.04105.

18. R. Quian Quiroga and H. Garcia. 2003. "Single-trial Event-related Potentials with Wavelet Denoising," *Clinical Neurophysiology 114*(2):376–390.

19. Jaakko Sarela and Harri Valpola. 2005. "Denoising Source Separation," *Journal of Machine Learning Research 6*(March):233–272.

20. Claude Shannon. 1956. "The Zero Error Capacity of a Noisy Channel." *IRE Transactions on Information Theory 2*(3):8–19.

21. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhut- dinov. 2014. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research 15*(1):1929–1958.

22. Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. "Show and Tell: A Neural Image Caption Generator." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 31–56). June 7–12, Boston, MA.

23. Alle Meije Wink and Jos BTM Roerdink. 2004. "Denoising Functional MR Images: A Comparison of Wavelet Denoising and Gaussian Smoothing," *IEEE Transactions on Medical Imaging 23*(3):374–387.

24. Mattew D/ Zeiler and Rob Fergus. 2013. "Visualizing and Understanding Convolutional Networks." Cornell University Library. Computing Research Repository (CoRR). abs/1311.2901, 2013.

25. Matthew D. Zeiler, Dilip Krishnan, Graham W. Taylor, and Rob Fergus. 2010. "Deconvolutional Networks." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2528–2538). June 8–13, San Francisco, CA.

26. Junbo Zhao, Michael Mathieu, Ross Goroshin, and Yann LeCun. 2015. "Stacked What-Where Auto-encoders." Cornell University Library. Computing Research Repository (CoRR). abs/1506.02351.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-01-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
September 2016	Final	

4. TITLE AND SUBTITLE

Radio Model-free Noise Reduction of Radio Transmissions with Convolutional Autoencoders

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHORS

Benjamin Migliori
Daniel Gebhardt
Logan Straatemeier
Michael Walton

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

SSC Pacific
53560 Hull Street
San Diego, CA 92152–5001

8. PERFORMING ORGANIZATION REPORT NUMBER

TR 3047

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

SSC Pacific
Naval Innovative Science and Engineering (NISE) Program
53560 Hull Street
San Diego, CA 92152–5001

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Aproved for public release.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This technical report presents the findings of an experiment to evaluate the effectiveness of our technique for improving the accuracy of identifying which type of digital modulation is present in a sample of radio signal data. We use a convolutional neural network (CNN) to identify the modulation type from raw digitized radio signal input. The CNN is trained using our technique of dataset augmentation, which applies a transformation specific to the sensory domain of radio (and potentially, closely related signal types). This augmentation simulates a receiver's clock offset or error.

Digital radio signal receivers will have a clock frequency slightly different than the transmitter, even if each is tuned to the "same" frequency. This is usually accounted for in the receiver design, referred to as carrier clock recovery, since it is designed for a known signal type. Our method is to apply varying amounts of clock frequency offset to a training dataset, and use it to train the machine learning algorithm (in this case, a CNN). The trained CNN model is compared to a baseline model in which no clock offset was used during training.

Classification performance increases to nearly 100% when trained with frequency offset, compared to the baseline of 58%. Two real-world signals were captured from car remote keyless entry fobs. These signals contain an unknown receiver clock offset. The network trained with our method classified nearly 100% of the samples correctly, while the baseline network did not correctly identify the on-off keying (OOK) modulation. A recommended action is to further investigate dataset augmentation, especially in the domain of radio signals. This domain could benefit from very specific, but very useful, transforms to further improve performance of machine learning techniques.

15. SUBJECT TERMS

digital modulation; radio signal data; convolutional data network; dataset augmentation; clock offset; machine learning algorithm; frequency offset; on-off keying modulation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Benjamin Migliori
U	U	U	U	20	19B. TELEPHONE NUMBER *(Include area code)* (619) 553-9269

INITIAL DISTRIBUTION

84300	Library	(1)
56120	D. Gebhardt	(1)
56140	L. Staatemeir	(1)
56150	B. Migliori	(1)
56150	M. W. Walton	(1)

Defense Technical Information Center
Fort Belvoir, VA 22060–6218 (1)

Approved for public release.

SSC Pacific
San Diego, CA 92152-5001